REMARKS

In the Office Action, the Examiner indicated that Claims 1 through 20 are pending in the application and the Examiner rejected all claims.

Claim Rejections, 35 U.S.C. §102 and §103

On page 2 of the Office Action, the Examiner rejected Claims 1-2, 4-9, 11-16, and 18-20 under 35 U.S.C. §102(e) as being anticipated by "Secure Java Class Loading", IEEE Internet Computing, November/December 1998, Pages 56-61 by Gong ("Gong"). On page 8 of the Office Action, the Examiner rejected Claims 3, 10 and 17 under 35 U.S.C. §103(a) as being unpatentable over Gong in view of U.S. Application Publication No. 2003/0115218 to Bobbitt et al. ("Bobbitt").

The Present Invention

The present invention is a method, system, and computer program product for compiling Java code. In accordance with the present invention, Java code that references classes residing in a workspace can be complied. In accordance with the present invention, a workspace identifier is placed within the classpath to indicate the location of the referenced classes that may reside within a workspace. Specifically, Claim 1 recites: "1) determining if a referenced class file is located in a workspace; 2) locating the class file; 3) accessing the class file; and 4) returning the class file data to the compiler." (lines 3-6) In accordance with a preferred embodiment, the files on a web site are serviced using a file database, and a class file is allocated to a workspace by creating an additional entry in the file database. The class file is invoked by the compiler through the database.

<u>"Secure Java Class Loading", IEEE Internet Computing, Nov/Dec 1998, Pages 56-61 by Gong</u>

"Secure Java Class Loading", IEEE Internet Computing, November/December 1998, Pages 56-61 by Gong ("Gong") teaches dynamic class loading as a feature of the Java virtual machine. Gong defines several unique characteristics of class loading, including loading classes on demand, on a just-in-time basis. Second, dynamic class loading maintains type safety of the Java virtual machine by adding link-time checks, which replace other runtime checks. Additionally, dynamic class loading provides programmers with additional functionality, including defining their own classes, and the ability to utilize class loaders to provide separate name spaces for various software components. The Examiner acknowledges that Gong fails to disclose an indicator comprising a signature string, a user ID, a project ID, and a workspace name.

<u>U.S. Application Publication No. 2003/0115218 to Bobbitt et al.</u>

U.S. Application Publication No. 2003/0115218 to Bobbitt et al. ("Bobbitt") teaches a virtual file system. The system architecture enables a plurality of underlying file systems running on various file servers to be "virtualized" into one or more "virtual volumes" that appear as a local file system to clients that access the virtual volumes. The system also enables the storage spaces of the underlying file systems to be aggregated into a single virtual storage space, which can be dynamically scaled by adding or removing file servers without taking any of the file systems offline and in a manner transparent to the clients. The Examiner relies on Bobbitt for an alleged teaching of an indicator comprising a signature string, a user ID, a project ID, and a workspace name.

<u>The Cited Prior Art Does Not Anticipate the Claimed Invention</u>

The MPEP and case law provide the following definition of anticipation for the purposes of 35 U.S.C. §102:

> "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." MPEP §2131 citing *Verdegaal Bros. v. Union Oil Company of California*, 814 F.2d 628, 631, 2 U.S.P.Q. 2d 1051, 1053 (Fed. Cir. 1987)

<u>The Examiner Has Not Established a *Prima Facie* Case of Anticipation</u>

As noted above, the present claimed invention includes determining if a referenced class is located in a workspace, locating the class file in the workspace, accessing the class file, and returning the class file data to a compiler. By following these steps, class files do not need be stored in a standard designated workspace, rather class files can be spread across a storage system.

Gong teaches a Java virtual machine (JVM) using a classpath and ClassLoaders to execute a bytecode in the JVM, or essentially, running Java applications. This differs greatly from the presently claimed invention. The present claimed invention is not concerned with the execution of bytecodes, but rather the compiling of class file data. Specifically, the present claimed invention locates a class file, accesses a class file, and <u>returns the class file to a compiler for further processing.</u> Gong fails to teach returning any accessed class files to a compiler. The Examiner relies on a quotation from Gong referring to a browser loading applets from multiple locations using separate class loaders, then states "the preceding text excerpt clearly indicates that the class file is eventually loaded, which indicates that it is accessed and then returned to the compiler" (page 4 of the Office Action). Applicants respectfully disagree with this assertion. First, a browser executing a JVM to run a java bytecode is not the same as a compiler. The browser is merely functioning as a means to execute the bytecode.

There is no additional compiling of any code once it is passed to the JVM running in the browser. Additionally, Gong provides no teaching of returning any accessed class file data to a compiler. Accordingly, each of the independent claims, and all claims depending therefrom, patentably define over Gong and are in condition for allowance. The Examiner is respectfully requested to reconsider and withdraw the rejections of Claims 1-2, 4-9, 11-16, and 18-20 under 35 U.S.C. §102(e) as being anticipated by Gong.

Conclusion

The present invention is not taught or suggested by the prior art. Accordingly, the Examiner is respectfully requested to reconsider and withdraw the rejection of the claims. An early Notice of Allowance is earnestly solicited.

The Commissioner is hereby authorized to charge any fees associated with this communication to Deposit Account No. 09-0461.

Respectfully submitted,

April 11, 2007
Date

/John R. Brancolini/
John R. Brancolini
Registration No. 57,218

SYNNESTVEDT & LECHNER LLP
1101 Market Street
Suite 2600
Philadelphia, PA 19107
Telephone: (215) 923-4466
Facsimile: (215) 923-2189

S:\I\IBM\IBM RALEIGH RSW\PATENTS\P27088 USA\DRAFTS\REPLY TO OA OF 01112007.DOC